How2Sec security

Building your knowledge,
making your way!

Visão

Com a crescente demanda sobre Tecnologias, percebemos que muitas pessoas apesar de buscarem informações, não possuem fontes que queiram realmente passar o conhecimento da maneira como ela deve ser, livre e com embasamento técnico que permita ser aplicado e utilizado quando necessário, além de serem testados em sua criação, tornando esta informação útil e confiável.

Missão

O Laboratório foi criado com a intenção de buscar e disseminar o conhecimento de uma maneira clara e objetiva, de forma gratuita, auxiliando na evolução dos membros e da sociedade na qual estas informações são compartilhadas, buscando o crescimento de todos os envolvidos nesta criação de valores.

Autor: Wellington Silva a.k.a w3ll

Licença

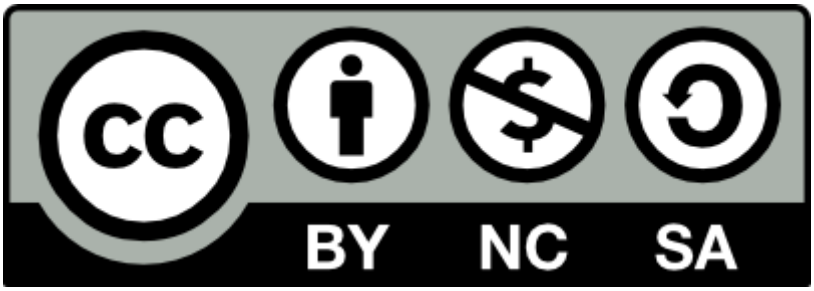

Figura 01 – Licença Criative Commons – by-nc-as

Esta licença permite que outros remixem, adapte, e criem obras derivadas sobre a obra original, desde que com fins não comerciais e contanto que atribuam crédito ao autor e licenciem as novas criações sob os mesmos parâmetros. Outros podem fazer download ou redistribuir a obra da mesma forma que na licença anterior, mas eles também podem traduzir, fazer remixes e elaborar novas histórias com base na obra original. Toda nova obra feita a partir desta deverá ser licenciada com a mesma licença, de modo que qualquer obra derivada, por natureza, não poderá ser usada para fins comerciais.

This license lets other remix, tweak, and build upon your work non-commercially, as long as they credit you and license their new creations under the identical terms.

Para maiores informações sobre o método de licenciamento acesse os seguintes sites:

Brasil:
http://creativecommons.org.br/as-licencas/
http://creativecommons.org/licenses/by-nc-sa/3.0/br/

Internacional:
http://creativecommons.org/licenses/by-nc-sa/3.0/
http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode

Autor: Wellington Silva a.k.a w3ll

1 – Proteção do Sistema de Arquivos

Aqui iremos abordar as melhores práticas para deixar nosso sistema de arquivos mais seguro. Para isso, iremos limitar o uso de **SUID Bit**, que pode implicar em falhas de segurança em binários e scripts. Além disso, iremos proteger o sistema de arquivos contra códigos arbitrários/maliciosos, tanto na montagem dos file systems como em seus atributos. Iremos proteger nossos diretórios de logs, diretórios temporários e estáticos.

2 – SUID Bit, SGID Bit e Stick Bit

No Linux temos algumas permissões especiais para nos ajudar em facilidades e segurança, além da tradicional **UGO** (**User, Group** and **Other**).

Nas opções do **UGO**, temos as seguintes permissões:

Binário	Octal	Representação	Descrição
000	0	_ _ _	Sem Permissão.
001	1	_ _ x	Permissão de Execução.
010	2	r _ _	Permissão de Leitura.
011	3	r _ x	Permissão de Leitura e Execução.
100	4	_ w _	Permissão de Escrita.
101	5	_ w x	Permissão de Escrita e Execução.
110	6	r w _	Permissão de Leitura e Escrita.
111	7	r w x	Permissão de Leitura, Escrita e Execução.

Tabela 01 – Permissões UGO

Como podemos observar elas não são muito flexíveis. Com as permissões especiais, temos as seguintes possibilidades:

Permissão	Valor Octal	Representação	Descrição
SUID Bit	4000	--s --- --- --S --- ---	Esse bit só pode ser aplicado em arquivos executáveis. Assim, o binário é executado com as permissões do dono.
SGID Bit	2000	--- --s --- --- --S ---	O SGID quando é aplicado em um arquivo executável, funciona de forma parecida com o SUID Bit, mas é usada a permissão do grupo do arquivo. Além disso, esse bit pode ser aplicado em diretórios, quando isso acontece, tudo que for criado dentro desse diretório terá o grupo dele, ou seja, se o diretório pai tem o SGID Bit ativado, tudo que for criado dentro dele terá o seu grupo e não o do usuário que criou.
Stick Bit	1000	--- --- --t --- --- --T	Permissão que impede que um usuário apague o arquivo ou diretório criado por outro usuário, mesmo que no diretório pai tenha permissão para excluir, apenas o usuário dono ou o root conseguirá. Temos como exemplo de uso o **/tmp**.

Tabela 02 – Permissões Especiais

Autor: Wellington Silva a.k.a w3ll

Como podemos ver o mais preocupante é o **SUID Bit**, já que, o binário será executado como usuário dono.

Por isso, o uso do **SUID Bit** deve ser limitado, já que isso pode implicar em falha de segurança. O **SUID Bit** quando aplicado em arquivos executáveis (já que o **SUID Bit** aplicado em diretório não tem efeito algum), faz com que este arquivo seja executado com as permissões do usuário dono do arquivo. Por exemplo, um usuário que execute um script que necessita gravar o log na pasta **/var/log** e este usuário não tem as permissões necessárias, poderemos usar o **SUID Bit** para que o usuário execute o script como se fosse o dono do script (como **root**, por exemplo) o qual tem as permissões necessárias de escrita no devido diretório.

Com base nisto, limitaremos o uso de **SUID Bit**. Iremos tirar todos os **SUID Bit** com exceção do comando **su,** para que os usuários de suporte possam utilizá-lo para entrar com as credenciais de **root**.

Para facilitar nossa vida, vamos criar um script para automatizar o processo e salvar o resultado, onde possamos arquivar o que foi feito (Auditado).

Vamos criar um script chamado **nosuid** em **/usr/local/sbin** e atribuir a permissão de **700** para o **root**.

```
root@fusion:~# > /usr/local/sbin/nosuid
root@fusion:~# chmod 700 /usr/local/sbin/nosuid
root@fusion:~# vi /usr/local/sbin/nosuid

#!/bin/bash

### CORES
amarelo="\e[33;1m"
azul="\e[34;1m"
verde="\e[32;1m"
vermelho="\e[31;1m"
fim="\e[m"

if [ `whoami` != "root" ]; then
        echo -e "$vermelho EXECUTE SOMENTE COMO ROOT$fim"
        exit 1
fi

if [ -z "$1" ] || [ "$1" == " " ]; then
        echo -e "$vermelho AVISO: Voce devera indicar o caminho e um nome para o
arquivo"
        echo -e " para posterior auditoria\012"
        echo -e " =========="
        echo -e " = EXEPLO ="
        echo -e " =========="
        echo -e " # nosuid /root/auditoria/nosuid.audit$fim"
        exit 1
fi

clear

echo -e "$amarelo AGUARDE... PROCURANDO POR APLICACOES COM SUID BIT
ATIVADO...\012$fim"

rel()
{
        echo -e "Aplicacoes com SUID Bit ativado...\012" >> /tmp/"$$".suid
        echo -e "\012\012" >> /tmp/"$$".suid
```

Autor: Wellington Silva a.k.a w3ll

```
        echo -e "\012\011****\011Os arquivos abaixo foram alterados\011****\012"
>> /tmp/"$$".nosuid
        echo -e "\012" >> /tmp/"$$".nosuid
        date >> /tmp/"$$".nosuid
        echo -e "\012" >> /tmp/"$$".yessuid
        echo -e "\012\011****\011Os arquivos abaixo NAO foram
alterados\011****\012" >> /tmp/"$$".yessuid
        echo -e "\012" >> /tmp/"$$".yessuid
        date >> /tmp/"$$".yessuid
        echo -e "\012" >> /tmp/"$$".yessuid
}
rel

for SUID in `find / -perm -4000 2> /dev/null` ;do
        echo -e " DESEJA$vermelho REMOVER$fim O SUID BIT DO$vermelho $SUID$fim?
(S -> PARA SIM): "
        echo "$SUID" >> /tmp/"$$".suid
        read ACAO

        case $ACAO in
                S|s)
                        chmod u-s $SUID
                        echo "$SUID" >> /tmp/"$$".nosuid
                        echo -e "$verde SUID BIT REMOVIDO COM SUCESSO$fim"
                ;;
                *)
                        echo "$SUID" >> /tmp/"$$".yessuid
                        echo -e "$azul SUID NAO FOI REMOVIDO$fim"
                ;;
        esac
done

echo -e "$amarelo AGUARDE... GERANDO O RELATORIO...$fim\012"

sleep 2

cat /tmp/"$$".suid >> "$1"
cat /tmp/"$$".nosuid >> "$1"
cat /tmp/"$$".yessuid >> "$1"
cat /tmp/"$$".nosuid
cat /tmp/"$$".yessuid

rm -rf /tmp/"$$".suid
rm -rf /tmp/"$$".nosuid
rm -rf /tmp/"$$".yessuid

exit 0
```

Pronto, com o script feito vamos executar e remover as permissões especiais de **SUID Bit**.

```
root@fusion:~# nosuid
 AVISO: Voce devera indicar o caminho e um nome para o arquivo
 para posterior auditoria

 ==========
 = EXEPLO =
 ==========
 # nosuid /root/auditoria/nosuid.audit
root@fusion:~# nosuid /root/auditoria/nosuid.audit
```

Autor: Wellington Silva a.k.a w3ll

```
root@fusion:~# nosuid /root/auditoria/nosuid.audit
 AGUARDE... PROCURANDO POR APLICACOES COM SUID BIT ATIVADO...

 DESEJA REMOVER O SUID BIT DO /bin/mount? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /bin/umount? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /bin/ping6? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /bin/su? (S -> PARA SIM):
n
 SUID NAO FOI REMOVIDO
 DESEJA REMOVER O SUID BIT DO /bin/ping? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /usr/lib/eject/dmcrypt-get-device? (S -> PARA
SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /usr/lib/pt_chown? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /usr/lib/openssh/ssh-keysign? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /usr/bin/newgrp? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /usr/bin/chsh? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /usr/bin/chfn? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /usr/bin/gpasswd? (S -> PARA SIM):
s
 SUID BIT REMOVIDO COM SUCESSO
 DESEJA REMOVER O SUID BIT DO /usr/bin/passwd? (S -> PARA SIM):
n
 SUID NAO FOI REMOVIDO
 AGUARDE... GERANDO O RELATORIO...

Aplicacoes com SUID Bit ativado...

        ****    Os arquivos abaixo foram alterados      ****



Wed Aug  1 20:17:13 BRT 2012
/bin/mount
/bin/umount
/bin/ping6
/bin/ping
/usr/lib/eject/dmcrypt-get-device
/usr/lib/pt_chown
```

Autor: Wellington Silva a.k.a w3ll

```
/usr/lib/openssh/ssh-keysign
/usr/bin/newgrp
/usr/bin/chsh
/usr/bin/chfn
/usr/bin/gpasswd


        ****    Os arquivos abaixo NAO foram alterados  ****



Wed Aug  1 20:17:13 BRT 2012


/bin/su
/usr/bin/passwd
root@fusion:~#
```

Neste script nós validamos as opções passadas junto com a execução do script **nosuid** para que sejam auditadas as alterações feitas. Após isso, seremos informados qual arquivo foi encontrado no sistema com o **SUID Bit** configurado e questionados se gostaríamos de tirar o **SUID Bit**. Digitamos **S** ou **s** para responder que **sim** e qualquer outra tecla para não. Por fim um relatório será gerado no caminho passado no script e nos mostrará o resultado na tela.

Recomenda-se que apenas os seguintes arquivos tenham permissão de **SUID Bit**, porém cada caso é um caso, e vale apena estudar o contexto de uso do servidor. Por exemplo, em um firewall não costumamos criar muitos usuários então para que deixar o **SUID Bit** nos arquivos abaixo?

- ***/usr/sbin/userhelper***
- ***/usr/sbin/suexec***
- ***/usr/bin/passwd***
- ***/usr/bin/sudo***
- ***/usr/bin/crontab***
- ***/bin/ping***
- ***/sbin/pam_timestamp_check***
- ***/sbin/unix_chkpwd***
- ***/sbin/unix2_chkpwd***
- ***/sbin/pwdb_chkpwd***

Observação: Podemos criar um script semelhante para remover as permissões de **SGID Bit**, inibindo que outros possam executar algum binário com o **GID (Group IDentifier)** do usuário dono do binário.

Autor: Wellington Silva a.k.a w3ll

3 – Proteção do FileSystem

Sempre que instalamos um sistema operacional Linux, devemos seguir a **FHS (FileSystem Hierarchy Standard)** e separar as partições que são compartilhadas para que possamos usar opções de montagem diferentes entre elas, além de nos proporcionar uma maior segurança.

Nós retiramos o **SUID Bit** no tópico anterior, porém isso, não resolve todo o problema. Olhe um exemplo de um diretório compartilhado que todos usuários têm acesso, e é o preferido dos invasores por ter essa facilidade. O ***/tmp***.

```
root@fusion:~# cp /bin/bash /tmp
root@fusion:~# cp /bin/sh /tmp
root@fusion:~# chmod u+s /tmp/bash
root@fusion:~# chmod u+s /tmp/sh
root@fusion:~# su - W3ll
root@fusion:~$ cd /tmp/
root@fusion:~$ ls
bash  lost+found  sh
W3ll@fusion:~$ ./bash
bash-4.1$ id
uid=1001(W3ll) gid=1001(W3ll) groups=1001(W3ll)
bash-4.1$ exit
exit

W3ll@fusion:~$ ./sh
# id
uid=1001(W3ll) gid=1001(W3ll) euid=0(root) groups=0(root),1001(W3ll)
# exit
$ exit
logout
root@fusion:~# mount -o remount,nosuid /tmp
root@fusion:~# su - W3ll
W3ll@fusion:~$ cd /tmp
W3ll@fusion:~$ ls -la
total 912
drwxrwxrwt  3 root root   4096 Jul 31 02:17 .
drwxr-xr-x 21 root root   4096 Jul 19 22:33 ..
-rwsr-xr-x  1 root root 811156 Jul 31 02:03 bash
drwx------  2 root root  16384 Jul 19 22:30 lost+found


-rwsr-xr-x  1 root root  84144 Jul 31 02:03 sh
   ^
   |
   +------- SUID Bit atribuido

W3ll@fusion:~$ ./bash
-su: ./bash: Permission denied
W3ll@fusion:~$ ./sh
-su: ./sh: Permission denied
W3ll@fusion:~$ exit
logout

root@fusion:~#
```

Como podemos ver o perigo de ter um arquivo com **SUID Bit**, executamos como usuário comum e nos tornamos **root** sem ser incomodado por uma senha.

Autor: Wellington Silva a.k.a w3ll

Para inibirmos isso, devemos definir alguma política de segurança em relação aos sistemas de arquivos montados. Isso pode ser feito utilizando algumas opções do comando **mount**, como foi feito no exemplo, atribuindo o **nosuid** na remontagem da partição ***/tmp***.

No arquivo **/etc/fstab**, iremos fazer alguns ajustes, altere como o seguinte:

```
root@fusion:~# vi /etc/fstab

# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
proc            /proc           proc    defaults        0       0
# / was on /dev/sda5 during installation
UUID=40ef5be0-9a5e-44c7-8114-d0cc553df50a /               ext3    errors=remount-
ro 0       1
# /boot was on /dev/sda1 during installation
UUID=30760ba9-dc35-4e83-b286-5fa9ba3799a6 /boot           ext3
defaults,noexec,nosuid,nodev        0       2
# /home was on /dev/sda7 during installation
UUID=a3ca48cb-f050-4ddb-b934-d260df4b911a /home           ext3
defaults,noexec,nosuid,nodev        0       2
# /srv was on /dev/sda12 during installation
UUID=17fec127-3da1-4cc7-99f2-3918c8815e06 /srv            ext3
defaults,noexec,nosuid,nodev        0       2
# /tmp was on /dev/sda8 during installation
UUID=9d9621b6-4cf7-455f-8574-fbff59e51ffa /tmp            ext3
defaults,noexec,nosuid,nodev        0       2
# /usr was on /dev/sda9 during installation
UUID=25d4ec5f-f140-4033-8593-913722cfbc99 /usr            ext3
defaults,ro,nodev       0       2
# /var was on /dev/sda10 during installation
UUID=0400b46b-3392-4d03-ae8d-758fec52b0d1 /var            ext3
defaults,noexec,nosuid,nodev        0       2
# /var/log was on /dev/sda11 during installation
UUID=a06466a1-ac20-4177-b949-1f564091805d /var/log        ext3
defaults,noexec,nosuid,nodev        0       2
# swap was on /dev/sda6 during installation
UUID=a8223485-5ddd-4046-a164-496a899b1a35 none  swap    sw              0       0
/dev/scd0       /media/cdrom0   udf,iso9660 user,noauto      0       0
/dev/fd0        /media/floppy0  auto    rw,user,noauto0       0

"/etc/fstab" 28L, 1886C written
root@fusion:~# mount -a
root@fusion:~# mount
/dev/sda5 on / type ext3 (rw,errors=remount-ro)
tmpfs on /lib/init/rw type tmpfs (rw,nosuid,mode=0755)
proc on /proc type proc (rw,noexec,nosuid,nodev)
sysfs on /sys type sysfs (rw,noexec,nosuid,nodev)
udev on /dev type tmpfs (rw,mode=0755)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
devpts on /dev/pts type devpts (rw,noexec,nosuid,gid=5,mode=620)
/dev/sda1 on /boot type ext3 (rw)
/dev/sda7 on /home type ext3 (rw)
/dev/sda12 on /srv type ext3 (rw)
/dev/sda8 on /tmp type ext3 (rw,noexec,nosuid,nodev)
```

```
/dev/sda9 on /usr type ext3 (rw)
/dev/sda10 on /var type ext3 (rw)
/dev/sda11 on /var/log type ext3 (rw)
root@fusion:~#
```

Com isso garantimos que toda vez que nosso sistema for reiniciado ele estará com suas partições protegidas.

Também devemos saber o que cada opção usada faz:

- **noexec** → A opção noexec especifica que o sistema de arquivos não pode conter binários executáveis.
- **nosuid** → A opção nosuid especifica que um sistema de arquivos não pode conter arquivos com **SUID Bit** habilitado.
- **nodev** → A opção nodev especifica que o sistema de arquivo não pode conter dispositivos especiais.

Os pontos de montagens que alteramos devem ser separados para proteger o sistema contra esgotamento e suas permissões diferenciadas, além de terem as seguintes funções:

- **/tmp** → O diretório ***/tmp*** é projetado unicamente para o armazenamento de arquivos temporários.
- **/var** → O diretório ***/var*** é usado por daemons e outros serviços do sistema para armazenar dados dinâmicos temporariamente.
- **/var/tmp** → O ***/var/tmp*** é normalmente um diretório standalone no sistema de arquivos do **/var**.
- **/var/log** → O diretório **/var/log** é usado pelos serviços do sistema para armazenar dados de auditoria.
- **/home** → O diretório ***/home*** é usado para armazenamento em disco pelos usuários locais.

Vamos criar um script para desativar esta configuração quando for necessária alguma manutenção no sistema operacional

```
root@fusion:~# > /usr/local/sbin/open-fs
root@fusion:~# chmod 700 /usr/local/sbin/open-fs
root@fusion:~# vi /usr/local/sbin/open-fs

#!/bin/bash

### CORES
amarelo="\e[33;1m"
azul="\e[34;1m"
verde="\e[32;1m"
vermelho="\e[31;1m"
fim="\e[m"

if [ `whoami` != "root" ]; then
        echo -e "$vermelho EXECUTE SOMENTE COMO ROOT$fim"
        exit 1
fi

start_openfs()
{
```

```
        mount -o remount,rw,nodev,noexec,nosuid /boot
        mount -o remount,rw,nodev,noexec,nosuid /home
        mount -o remount,rw,nodev,noexec,nosuid /tmp
        mount -o remount,rw,nodev,noexec,nosuid /var
        mount -o remount,rw,nodev,noexec,nosuid /var/log
        mount -o remount,ro,nodev /usr

        clear

        echo -e "\012"

        echo -e "$verde#================================================#$fim"
        echo -e "$verde#\011 PARTICOES SEM PERMISSAO DE EXECUCAO\011#$fim"
        echo -e "$verde#================================================#$fim"

        echo –e “\012”

        mount
}

stop_openfs()
{
        mount -o remount,rw,exec /boot
        mount -o remount,rw,exec /home
        mount -o remount,rw,exec /tmp
        mount -o remount,rw,exec /var
        mount -o remount,rw,exec /var/log
        mount -o remount,rw /usr

        clear

        echo -e "\012"

        echo -e
"$vermelho#================================================#$fim"
        echo -e "$vermelho#\011 PARTICOES COM PERMISSAO DE EXECUCAO\011#$fim"
        echo -e
"$vermelho#================================================#$fim\012"

        mount
}

case $1 in
        start)
                start_openfs
        ;;
        stop)
                stop_openfs
        ;;
        *)
                echo -e "$amarelo AVISO: Use $0 {start|stop}$fim"
                echo -e "\012"
                echo -e " EXEMPLO"
                echo -e " ======="
                echo -e " # open-fs start"
                echo -e "\012"
                exit 1
        ;;
```

```
esac

exit 0
```

Com isso protegemos nosso sistema de arquivos contra códigos arbitrários e maliciosos.

Caso precisemos realizar alguma manutenção no sistema, basta executarmos o script criado acima com o parâmetro **stop**. Para voltarmos usamos o parâmetro **start**.

Como nosso objetivo é proteger os sistemas de arquivos, devemos nos atentar que para instalar algum pacote pré-compilado com os aplicativos **apt-get** e **aptitude** as partições ***/tmp*** e ***/var*** deverão ter permissões de **exec** e a partição ***/usr*** deverá ter permissão de **rw**.

```
root@fusion:~# aptitude install c-essential

-==[ Resumido ]==--

Get:5 http://ftp.us.debian.org/debian/ squeeze/main libc6-dev i386 2.11.3-3
[4,799 kB]

-==[ Resumido ]==--

Selecting previously deselected package libc-dev-bin.
Unpacking libc-dev-bin (from .../libc-dev-bin_2.11.3-3_i386.deb) ...
dpkg: error processing /var/cache/apt/archives/libc-dev-bin_2.11.3-3_i386.deb (--
unpack):
 unable to securely remove '/usr/bin/mtrace.dpkg-new': Read-only file system
configured to not write apport reports

-==[ Resumido ]==--

Setting up man-db (2.5.7-8) ...
dpkg (subprocess): unable to execute installed post-installation script
(/var/lib/dpkg/info/man-db.postinst): Permission denied
dpkg: error processing man-db (--configure):
 subprocess installed post-installation script returned error exit status 2
Errors were encountered while processing:
 man-db

Current status: 6 updates [+6].
root@fusion:~#
```

Para resolver isso, vamos criar um arquivo em ***/etc/apt/apt.conf.d*** com o nome ***01remount,*** ficando desta forma:

```
root@fusion:~# > /etc/apt/apt.conf.d/01remount
root@fusion:~# vi /etc/apt/apt.conf.d/01remount

DPkg
{
        Pre-Invoke      { "mount -o remount,rw /usr" };
        Post-Invoke     { "mount -o remount,ro /usr" };
};

DPkg
{
        Pre-Invoke      { "mount -o remount,exec /var" };
        Post-Invoke     { "mount -o remount,noexec /var" };
};
```

```
DPkg
{
        Pre-Invoke      { "mount -o remount,exec /tmp" };
        Post-Invoke     { "mount -o remount,noexec /tmp" };
};

"/etc/apt/apt.conf.d/01remount" 17L, 311C written

#
```

Nota: Para gerenciadores com o **yum** e o **yast** você deve usar o script **open-fs** criado anteriormente.

Algumas vezes, poderá aparecer uma mensagem de erro dizendo que **"/usr busy"** (***/usr*** está ocupado). Isto acontece basicamente porque os arquivos estão sendo usados durante a atualização. Para confirma, podemos usar o aplicativo **lsof** da seguinte forma:

```
root@fusion:~# lsof +L1
```

Interrompa ou reinicie estes programas e execute manualmente o comando do **Post-Invoke**.

4 – Proteção nos Arquivos de Log do Sistema

Agora iremos proteger os logs do sistema usando o atributo de **append** do sistema de arquivos. Com isso, nossos logs não poderão ser apagados, reescritos e nem modificados, apenas novos registros poderão ser inclusos. Porém devemos ter alguns cuidados ao usar esse recurso pois, toda vez que desligamos o sistema, ele apaga o arquivo de log **dmesg** e toda vez em que ligamos ele recria este arquivo. Além de ser feita algumas modificações ou inclusão de novos arquivos de log em ***/var/log*** na instalação de alguns aplicativos. Fiquem tranquilos, pois iremos ver como contornar estas situações.

Primeiro vamos modificar o arquivo de **logrotate** (arquivo responsável por fazer o rotacionamento dos logs, segundo à política aplicada para armazenamento e descarte) em **/etc/cron.daily/logrotate**. Vamos alterar o arquivo:

```
root@fusion:~# vi /etc/cron.daily/logrotate
#!/bin/sh

test -x /usr/sbin/logrotate || exit 0
chattr -R -a /var/log
/usr/sbin/logrotate /etc/logrotate.conf
chattr -R +a /var/log


"/etc/cron.daily/logrotate" 6L, 133C written

root@fusion:~#
```

Com isso, toda vez que o log for rotacionado, será permitido a compactação dos logs anteriores e a criação dos novos logs.

Agora iremos precisar criar um script para remover e adicionar o atributo de **append** do diretório ***/var/log***.

Autor: Wellington Silva a.k.a w3ll

```
root@fusion:~# > /usr/local/sbin/log-attr
root@fusion:~# chmod 700 /usr/local/sbin/log-attr
root@fusion:~# vi /usr/local/sbin/log-attr

#!/bin/bash

### CORES
amarelo="\e[33;1m"
azul="\e[34;1m"
verde="\e[32;1m"
vermelho="\e[31;1m"
fim="\e[m"

if [ `whoami` != "root" ]; then
        echo -e "$vermelho EXECUTE SOMENTE COMO ROOT$fim"
        exit 1
fi

add_attr()
{
        chattr -R +a /var/log
        echo -e "$verde Atributo de APPEND adicionado aos arquivos de LOG$fim"
}

remove_attr()
{
        chattr -R -a /var/log
        echo -e "$vermelho Atributo de APPEND removido dos arquivos de LOG$fim"
}

case $1 in
        start)
                add_attr
        ;;
        stop)
                remove_attr
        ;;
        *)
                echo -e "$amarelo AVISO: Opcao desconhecida"
                echo -e " Use $0 {start|stop}$fim"
                exit 1
        ;;
esac

exit 0

"/usr/local/sbin/log-attr" 36L, 503C written

root@fusion:~#
```

Agora precisamos criar um script para desativar a opção de **append** quando formos desligar a máquina e adicionar o atributo quando ligarmos. Para isso, vamos criar o arquivo ***/etc/init.d/logattr***:

```
root@fusion:~# > /etc/init.d/logattr
root@fusion:~# chmod 700 /etc/init.d/logattr
root@fusion:~# vi /etc/init.d/logattr

#!/bin/bash
```

```
### BEGIN INIT INFO
# Provides:             logattr
# Required-Start:       $all
# Required-Stop:
# Should-Start:
# Should-Stop:
# Default-Start:        2 3 4 5
# Default-Stop:         0 1 6
# Short-Description:    Start the APPEND in LOG Files
### END INIT INFO

### CORES
amarelo="\e[33;1m"
azul="\e[34;1m"
verde="\e[32;1m"
vermelho="\e[31;1m"
fim="\e[m"

start_attr()
{
        /usr/local/sbin/log-attr start
}

stop_attr()
{
        /usr/local/sbin/log-attr stop
}

case $1 in
        start)
                start_attr
        ;;
        stop)
                stop_attr
        ;;
        *)
                echo -e "$amarelo Opcao desconhecida"
                echo -e " Use $0 {start|stop}$fim"
                exit 1
        ;;
esac

exit 0

"/etc/init.d/logattr" 45L, 634C written

root@fusion:~#
```

Agora adicionamos os links simbólicos dentro dos diretórios **rcX** com o seguinte comandos:

```
root@fusion:~# update-rc.d -f logattr defaults
update-rc.d: using dependency based boot sequencing
root@fusion:~#
```

Bom, lembrando que esse procedimento pode impactar na instalação de novas aplicações, pois, os novos aplicativos podem criar arquivos de log em /var/log, devemos

acrescentar as linhas: ***/usr/local/sbin/log-attr start*** e***/usr/local/sbin/log-attr stop*** dentro do arquivo ***/etc/apt/apt.conf.d/01remount*** para que fique assim:

```
root@fusion:~# vi /etc/apt/apt.conf.d/01remount

DPkg
{
        Pre-Invoke      { "mount -o remount,rw /usr" };
        Post-Invoke     { "mount -o remount,ro /usr" };
};

DPkg
{
        Pre-Invoke      { "mount -o remount,exec /var" };
        Post-Invoke     { "mount -o remount,noexec /var" };
};

DPkg
{
        Pre-Invoke      { "mount -o remount,exec /tmp" };
        Post-Invoke     { "mount -o remount,noexec /tmp" };
};

DPkg
{
        Pre-Invoke      { "/usr/local/sbin/log-attr stop" };
        Post-Invoke     { "/usr/local/sbin/log-attr start" };
};

"/etc/apt/apt.conf.d/01remount" [New] 5L, 110C written
root@fusion:~#
```

Com isso, quando usarmos o **apt-get** ou o **aptitude** será removido o atributo de **append** do ***/var/log*** e adicionado automaticamente quando finalizado.

5 – Proteção nos Arquivos de Sistema

Alguns arquivos que já vem em sistemas **UNIX-Like** não devem ser acessados ou alterados pelos usuários e por isso devem ser protegidos baseados no princípio do menor privilégio necessário.

São eles:

- **/** → O diretório raiz (/) é o nível mais alto na hierarquia do sistema de arquivos. Permissões incorretas nesse nível do file system possibilitam que usuários possam comprometer a disponibilidade ou integridade do sistema (por exemplo, através da remoção ou alteração não autorizada de arquivos necessários).
    - **chmod 755 /**
    - **chown root /**
    - **chgrp root /**
- **/etc** → O diretório ***/etc*** armazena arquivos de configuração do sistema operacional e de serviços, muitos dos quais possuem informações sensíveis a respeito do estado do sistema. Permissões incorretas no diretório ***/etc*** podem causar a indisponibilidade do sistema operacional através da alteração ou remoção não autorizada dos arquivos de configuração, além de possibilitar o acesso indevido a informações confidenciais.
    - **chmod 755 /etc**

    - **chown root /etc**
    - **chgrp root /etc**
- **/etc/crontab** → O arquivo ***crontab*** contém informações sobre trabalhos executados pelo cron. Permissões inadequadas podem permitir que usuários obtivessem informações do sistema, e possivelmente uma maneira de obter acesso privilegiado não autorizado.
    - **chown root:root /etc/crontab**
    - **chmod og-rwx /etc/crontab**
- **/etc/cron.hourly** → O diretório ***cron.hourly*** contém informações sobre trabalhos executados pelo **cron**. Permissões inadequadas, podem permitir que usuários obtivessem informações do sistema, e possivelmente uma maneira de obter acesso privilegiado não autorizado.
    - **chown root:root /etc/cron.hourly**
    - **chmod og-rwx /etc/cron.hourly**
- **/etc/cron.daily** → O diretório ***cron.daily*** contém informações sobre trabalhos executados pelo **cron**. Permissões inadequadas podem permitir que usuários obtivessem informações do sistema, e possivelmente uma maneira de obter acesso privilegiado não autorizado.
    - **chown root:root /etc/cron.daily**
    - **chmod og-rwx /etc/cron.daily**
- **/etc/cron.weekly** → O diretório ***cron.weekly*** contém informações sobre trabalhos executados pelo **cron**. Permissões inadequadas podem permitir que usuários obtivessem informações do sistema, e possivelmente uma maneira de obter acesso privilegiado não autorizado.
    - **chown root:root /etc/cron.weekly**
    - **chmod og-rwx /etc/cron.weekly**
- **/etc/cron.monthly** → O diretório ***cron.monthly*** contém informações sobre trabalhos executados pelo **cron**. Permissões inadequadas podem permitir que usuários obtivessem informações do sistema, e possivelmente uma maneira de obter acesso privilegiado não autorizado.
    - **chown root:root /etc/cron.monthly**
    - **chmod og-rwx /etc/cron.monthly**
- **/etc/cron.d** → O diretório ***cron.d*** contém informações sobre trabalhos executados pelo **cron**. Permissões inadequadas podem permitir que usuários obtivessem informações do sistema, e possivelmente uma maneira de obter acesso privilegiado não autorizado.
    - **chown root:root /etc/cron.d**
    - **chmod og-rwx /etc/cron.d**
- **/etc/passwd** → O arquivo /***etc/passwd*** armazena informações de contas dos usuários e serviços, como por exemplo, o shell utilizado, UID, GID, etc. Alterações não autorizadas neste arquivo, podem ocasionar o acesso indevido ao sistema ou até mesmo indisponibilidade do sistema.
    - **chmod 644 /etc/passwd**
    - **chown root /etc/passwd**
    - **chgrp root /etc/passwd**
- **/etc/shadow** → O arquivo ***/etc/shadow*** é responsável pelo armazenamento dos hashs das senhas dos usuários do sistema. Caso usuários mal intencionados tenham acesso indevido a este arquivo, poderiam ter acesso às senhas e, consequentemente, utilizar contas privilegiadas do sistema.
    - **chmod 630 /etc/shadow**
    - **chown root /etc/shadow**
    - **chgrp root /etc/shadow**
- **/etc/gshadow** → O arquivo ***/etc/gshadow*** é responsável pelo armazenamento dos hashs das senhas dos grupos do sistema. Caso usuários mal intencionados tenham acesso indevido a este arquivo, poderiam ter acesso às senhas e, consequentemente, utilizar contas privilegiadas do sistema.
    - **chmod 640 /etc/gshadow**
    - **chown root /etc/gshadow**
    - **chgrp root /etc/gshadow**

- **/etc/hosts.allow** → Permissões inadequadas no arquivo ***hosts.allow*** podem permitir a edição do mesmo por um possível atacante ou código malicioso, o que poderia facilitar acesso ao sistema.
    - **chmod 644 /etc/hosts.allow**
- **/etc/hosts.deny** → Permissões inadequadas no arquivo ***hosts.deny*** podem permitir a edição do mesmo por um possível atacante ou código malicioso, o que poderia facilitar acesso ao sistema.
    - **chmod 644 /etc/hosts.deny**
- **/etc/hosts** → O arquivo ***/etc/hosts*** associa endereços IP ao nome dos dispositivos de rede (hostnames). Permissões incorretas nesse arquivo podem permitir que nomes ou endereços fossem alterados e serviços ou aplicações que dependam dessas informações podem sofrer paralisação e consequente indisponibilidade.
    - **chmod 644 /etc/hosts**
    - **chown root /etc/hosts**
    - **chgrp root /etc/hosts**
- **/etc/ssh/sshd_config** → Uma vez que o serviço de SSH é responsável pelo acesso remoto, recomenda-se configurar adequadamente as permissões de acesso a seu arquivo de configuração, a fim de evitar que atacantes possam alterar seu conteúdo e assim abrir brechas de segurança para futuros ataques.
    - **chmod 600 /etc/ssh/sshd_conf**
    - **chown root /etc/ssh/sshd_conf**
    - **chgrp root /etc/ssh/sshd_conf**
- **/etc/motd**, **/etc/issue** e **/etc/issue.net** → A exibição de uma mensagem de advertência durante o login, tem como propósito deixar claro que o sistema em questão pertence à empresa e é de uso restrito para usuários autorizados, e que os acessos estarão sendo gravados em logs. Essas mensagens são armazenadas nos arquivos **/etc/issue**, ***/etc/issue.net***, ***/etc/motd***, dentre outros. Para evitar que algum usuário não autorizado modifique as mensagens, é recomendado que suas permissões sejam configuradas de forma a evitar acessos indevidos.
    - **chmod 644 /etc/motd**
    - **chown root /etc/motd**
    - **chgrp root /etc/motd**
    - **chmod 644 /etc/issue**
    - **chown root /etc/issue**
    - **chgrp root /etc/issue**
    - **chmod 644 /etc/issue.net**
    - **chown root /etc/issue.net**
    - **chgrp root /etc/issue.net**
- **/etc/inittab** → O arquivo de configuração **/etc/inittab** é responsável por determinar quais processos serão inicializados durante o boot e durante operações normais do sistema (por exemplo, o **runlevel** no qual o sistema será inicializado, configurações do processo de **shutdown** do sistema, etc). Alterações indevidas neste arquivo podem comprometer a disponibilidade e a integridade do sistema.
    - **chmod 640 /etc/inittab**
    - **chown root /etc/inittab**
    - **chgrp root /etc/inittab**
- **/etc/inetd.conf** → Caso seja utilizado, o arquivo ***/etc/inetd.conf*** será responsável pela inicialização dos serviços de rede do sistema. Convém que as permissões sejam configuradas de forma a impedir que usuários não autorizados possam ter acesso ao arquivo, o que poderia levar à paralisação de serviços e outros problemas de segurança.
    - **chmod 640 /etc/inetd.conf**
    - **chown root /etc/inetd.conf**
    - **chgrp root /etc/inetd.conf**
- **/etc/xinetd.conf** → Caso seja utilizado, o arquivo ***/etc/xinetd.conf*** será responsável pela inicialização dos serviços de rede do sistema. Convém que as

permissões sejam configuradas de forma a impedir que usuários não autorizados possam ter acesso ao arquivo, o que poderia levar à paralisação de serviços e outros problemas de segurança.
    - **chmod 640 /etc/xinetd.conf**
    - **chown root /etc/xinetd.conf**
    - **chgrp root /etc/xinetd.conf**
- **/etc/init.d** → O diretório ***/etc/init.d*** contém os scripts de inicialização do sistema que determinam quais serviços ou parâmetros de configuração serão inicializados no processo de boot. Permissões incorretas neste diretório tornam o sistema vulnerável a modificações não autorizadas que podem causar a indisponibilidade dos serviços ou a carga de serviços maliciosos.
    - **chmod -R 750 /etc/init.d**
    - **chown -R root /etc/init.d**
    - **chgrp -R root /etc/init.d**
- **/etc/securetty** → O arquivo ***/etc/securetty***, é responsável por configurar os terminais que o usuário root pode realizar o login. Antes de se efetivar o login, o sistema consulta este arquivo a fim de verificar se o terminal utilizado para efetuar o login é permitido. Recomenda-se configurar as permissões de acesso a este arquivo, a fim de evitar que atacantes possam alterar seu conteúdo e assim abrir brechas de segurança para futuros ataques.
    - **chmod 640 /etc/securetty**
    - **chown root /etc/securetty**
    - **chgrp root /etc/securetty**
- **/etc/group** → O arquivo ***/etc/group*** armazena informações de configuração dos grupos de usuários. Caso usuários não autorizados tenham acesso a este arquivo, eles poderão alterar as configurações dos grupos de forma a aumentar os seus privilégios no sistema. Além disso, alterações indevidas nesse arquivo podem ocasionar na paralisação de serviços.
    - **chmod 644 /etc/group**
    - **chown root /etc/group**
    - **chgrp root /etc/group**
- **/home/<user>** → O diretório home de cada usuário é especificado no arquivo ***/etc/passwd***. Esses diretórios contêm arquivos de inicialização e configuração individuais, dados pessoais, e em alguns casos, diretórios visíveis via web. Permissões incorretas nos diretórios home podem expor informações sensíveis a outros usuários e levar à perda da integridade e da disponibilidade dessas informações. Para sistemas onde há um serviço de e-mail funcionando, a permissão do home dos usuários deve ser **751** para que seja possível o recebimento de e-mails.
    - **chmod 700 /home/<usuario>**
    - **chown <usuario> /home/<usuario>**
    - **chgrp <grupo_do_usuario> /home/<usuario>**
- **~/.bash_history** → Quando uma shell é utilizado, é criado um arquivo denominado ***.bash_history*** no diretório home dos usuários que armazena todos os comandos digitados pelos mesmos. Recomenda-se configurar as permissões de acesso de forma que usuários não autorizados (que não sejam o owner) não possam visualizar copiar, editar ou remover este arquivo, visando evitar a perda de rastreabilidade dos comandos executados no contexto do usuário.
    - **find / -name .bash_history –ls**
    - **chmod 600 /home/<usuário>/.bash_history**
    - **chown <usuário> /home/<usuário>/.bash_history**
    - **chgrp <grupo_do_usuário> /home/<usuário>/.bash_history**
- **/var** → O diretório ***/var*** e seus subdiretórios podem armazenar informações como registros de logs, e-mails dos usuários, arquivos de bancos de dados, etc. O acesso de escrita neste diretório por parte de usuários não autorizados pode ocasionar a perda de integridade e disponibilidade destas informações.
    - **chmod 755 /var**
    - **chown root /var**
    - **chgrp root /var**

- **/var/log** → A análise crítica dos registros da auditoria permite detectar eventos relevantes para a segurança e são alvos preferenciais de hackers que tentam apagar evidências de suas ações sobre o sistema. Por este motivo, os logs do sistema precisam ser mantidos em local seguro, protegidos contra leituras e modificações não autorizadas.
    - **chmod 640 -R /var/log**
    - **chown root -R /var/log**
    - **chgrp log -R /var/log**
- **/sbin** → O diretório ***/sbin*** armazena vários programas e comandos, muitos dos quais essenciais para o bom funcionamento do sistema. Permissões inadequadas neste diretório tornam o sistema vulnerável a diversos ataques, podendo ter sua integridade comprometida ou até mesmo se tornar indisponível.
    - **chmod 755 /sbin**
    - **chown root /sbin**
    - **chgrp root /sbin**
- **/dev/mem** → O arquivo ***/dev/mem*** é um device file, o qual é uma imagem da memória principal do computador. Este dispositivo pode ser usado, por exemplo, para examinar (e até modificar) o conteúdo da memória do sistema. Permissões incorretas nesse arquivo podem ocasionar acessos indevidos a informações sensíveis e até ao próprio sistema bem como sua paralisação e consequente indisponibilidade.
    - **chmod 640 /dev/mem**
    - **chown root /dev/mem**
    - **chgrp root /dev/mem**
- **/tmp** → O diretório ***/tmp*** é responsável por armazenar arquivos temporários criados por aplicações ou usuários. Alterações não autorizadas podem comprometer o funcionamento de aplicações ou do próprio sistema ou permitir o acesso indevido a informações temporariamente armazenadas pelos usuários.
    - **chmod 640 /tmp**
    - **chown root /tmp**
    - **chgrp root /tmp**
- **/usr** → O diretório ***/usr*** pode conter códigos fontes e/ou binários do sistema, bibliotecas de desenvolvimento e documentação (por exemplo, man pages). Permissões incorretas no diretório ***/usr*** podem levar ao acesso não autorizado a informações ou a indisponibilidade de serviços do sistema.
    - **chmod 755 /usr**
    - **chown root /usr**
    - **chgrp root /usr**

Observação: Aqui não automatizamos porque deve ser levantando as permissões atuais e a real necessidade da aplicação dessas permissões, além de cada distro tratar de forma diferente alguns desses arquivos.

5.1 – Arquivos Sem Dono e Sem Grupo

Outra questão são os arquivos sem os atributos **owner** ou **group** definidos. Eles podem ser acessados por qualquer usuário, o que pode levar ao acesso indevido a informações sensíveis nele contidas. Convém que seja criado um procedimento de verificação periódica do sistema em busca de arquivos sem owner ou group, removendo-os se possível, ou alterando-se o atributo owner para que não fique indefinido.

```
root@fusion:~# find / \( -nouser -o -nogroup \) -ls

root@fusion:~# chown <nome_do_usuario> <arquivo> ou chown -R
<nome_do_usuario> <diretório>

root@fusion:~# chgrp <nome_do_grupo> <arquivo> ou chgrp -R <nome_do_grupo>
<diretório>
```

```
root@fusion:~#
```

Aqui temos apenas um exemplo de pesquisar o sistema atrás desses arquivos.

5.2 – Permissões Para Novos Arquivos com o UMASK

O valor **umask** define as permissões de acesso default de novos arquivos criados no sistema. Na maioria dos sistemas UNIX, o valor default do umask é **022**. Convém que este valor seja alterado para **027**, de forma que novos arquivos terão a permissão para arquivos mode **640** (rw- r-- ---) e para diretórios mode **750** (rwx r-x ---). Havendo necessidade, as permissões aplicadas pela umask poderão ser alteradas posteriormente com o comando **chmod**.

Para fazermos essa configuração basta esitar o arquivo ***/etc/profile*** e adicionar a seguinte linha:

```
root@fusion:~# vi /etc/profile

--==[ Resumido ]==--
umask 027

--==[ Resumido ]==--

"/etc/profile" 47L, 1065C written
root@fusion:~#
```

Além desse arquivo podemos fazer o mesmo nos seguintes arquivos:

- ***/etc/skel/.profile***
- ***.profile*** (de cada usuário)

Nota: Dependendo do interpretador de comandos (shell) utilizado no sistema, o arquivo de inicialização pode ter outro nome que não ***.profile***. Para o Bash, por exemplo, tal arquivo chama-se ***.bash_profile***.

5.3 – Permissões Para os Arquivos Binário SU

O **su** permite acesso ao sistema com privilégios de outros usuários (inclusive o root).

Embora o comando exija uma senha antes de conceder novos privilégios, recomenda-se que o mesmo seja restrito apenas a usuários autorizados, já que não há sentido em sua utilização por parte de usuários comuns que não executem tarefas administrativas no sistema. Além disso, o **su** não impõe restrições ao número de tentativas de acesso mal sucedidas e pode ser utilizado para tentativas de logon como root através de **Brute-Force**.

Para isso, podemos criar um grupo (que já temos criado, no artigo sobre acesso remoto, **supgrp**) e adicionar a esse grupo os usuário que podem executar o comando **su**.

```
root@fusion:~# addgroup supgrp

root@fusion:~# usermod –G supgrp W3ll

root@fusion:~# chmod 4550 /bin/su
```

```
root@fusion:~# chown root /bin/su

root@fusion:~# chgrp supgrp /bin/su

root@fusion:~# su w3ll

w3ll@fusion:~$ su - W3ll
bash: /bin/su: Permission denied

w3ll@fusion:~$ exit
exit

root@fusion:~# su - W3ll

W3ll@fusion:~$ su - w3ll
Password:

su: Authentication failure

W3ll@fusion:~$
```

5.4 – Atributo Imutável em Arquivos do Sistema

O atributo imutável (+i) evita que arquivos sejam alterados, apagados ou que links simbólicos sejam criados, o que reduz a possibilidade de erros acidentais e alterações não autorizadas. Por isso, convém que os arquivos de configuração críticos tenham esse atributo setado.

Alguns arquivos são:

- ***/etc/lilo.conf***
- ***/etc/grub.d/00_header***
- ***/etc/initd.conf***
- ***/etc/xinetd.conf***

Observação: Deve-se observar qual gerenciador de boot (**LLIO** ou **GRUB**), e qual super server (**Inetd** ou **Xinetd**) são utilizados no sistema, pois quando um é utilizado o controle não se aplica para o outro.

Nota: Adicionalmente, em sistemas onde a manutenção do arquivo de usuários não é muito frequente (por exemplo, web servers dedicados), convém habilitar o atributo (+i) nos arquivos ***/etc/passwd***, ***/etc/group*** e **/etc/shadow**.

Para isso, basta fazermos como no exemplo abaixo:

```
root@fusion:~# chattr +i /etc/xinetd.comn

root@fusion:~#
```

Observação: Para tirar o atributo de imutável basta trocar o sinal de **+** uo **-**.

5.5 – Arquivos de Inicialização dos Usuários

Arquivos de inicialização (***.profile***, ***.cshrc***, ***.login***, ***.mailrc***, ***.exrc***, ***.emacs***, ***.my.cnf***, etc) são responsáveis por carregar variáveis de ambiente e configurações do perfil dos usuários. O uso de permissões incorretas pode permitir que usuários mal intencionados comprometam a confidencialidade, disponibilidade, ou integridade destes arquivos.

Alguns deles são:

- ***.profile***
- ***.cshrc***
- ***.login***
- ***.mailrc***
- ***.exrc***
- ***.emacs***
- ***.my.cnf***
- etc

5.6 – Aplicativos Que Fazem Uso da Pilha TCP/IP

O acesso aos binários que implementam os serviços **TCP/IP** como o **telnet**, **ftp** e outros programas, facilita o acesso não autorizado a conteúdo disponível na Internet, que pode ser malicioso ou incompatível com a política de segurança da empresa (por exemplo, download de vídeos, arquivos mp3, etc). Além disso, os serviços **TCP/IP** podem ser utilizados para lançar ataques em sistemas locais ou remotos. Por estes motivos, convém que o acesso a estes arquivos seja restrito ao superusuário, ou quando estritamente necessário liberado somente para usuários autorizados.

Alguns deles são:

- **telnet**
- **ssh**
- **ftp**
- **nslookup**
- **dig**
- **netstat**
- **wget**
- **netcat**
- **lynx**
- **tftp**
- **nmap**
- **ip**
- **route**
- **rlogin**
- **rsh**
- **rcp**
- scp

Podemos criar um grupo para dar permissão de execução às exceções da seguinte forma:

```
root@fusion:~# for BIN in telnet ssh ftp nslookup dig netstat wget netcat
lynx tftp nmap ip route rlogin rsh rcp scp ;do
      chmod 750 $BIN;
      chown root $BIN;
      chgrp powerunix $BIN;
```

```
done

root@fusion:~#
```

5.7 – Aplicativos Compiladores e de Script

Compiladores e debuggers devem ser instalados somente quando seu uso for necessário e depois de utilizados, devem ser removidos, pois são programas potencialmente maliciosos. A existência destes utilitários nos sistemas Linux deve ser avaliada, pois, no caso do sistema ser comprometido, eles podem ser utilizados para introduzir novos programas e explorar outras vulnerabilidades, aumentando o nível de comprometimento do sistema. Caso a existência destes programas seja necessária, as permissões dos compiladores (**cc**, **gcc**, **perl**, **perlcc**, **make**, **javac**, **nasm** e outros) e dos debuggers (**gdb**, **sdb**, **strace**, **truss** e outros) devem ser configuradas de forma a permitir acesso apenas para usuários autorizados, além do superusuário root.

Podemos criar um grupo para dar permissão de execução às exceções da seguinte forma:

```
root@fusion:~# for BIN in cc gcc perl perlc make javac python nasm gdb sdb
strace truss ;do
      chmod 750 $BIN;
      chown root $BIN;
      chgrp devunix $BIN;
done

root@fusion:~#
```

5.8 – Aplicativos de Monitoração da Pilha TCP/IP

Ferramentas de rede (**tcpdump**, **ethereal**, **iptraf**, **nmap**, **netcat** e outros) potencialmente maliciosas, devem ser instaladas somente quando seu uso for necessário e depois de utilizadas, devem ser removidas. São programas potencialmente maliciosos. A existência dessas ferramentas em um sistema deve ser avaliada, pois no caso do sistema ser comprometido, elas podem ser utilizadas para capturar tráfego, scannear portas, ou explorar outras vulnerabilidades, aumentando o nível de comprometimento do sistema. Caso a existência de alguma ferramenta de rede seja necessária ao sistema, suas permissões devem ser configuradas de forma a permitir acesso apenas para usuários autorizados, além do superusuário root.

Podemos criar um grupo para dar permissão de execução às exceções da seguinte forma:

```
root@fusion:~# for BIN in tcpdump ethereal iptraf nmap nc ;do
      chmod 750 $BIN;
      chown root $BIN;
      chgrp devunix $BIN;
done

root@fusion:~#
```

5.9 – Aplicativos Que Desligam/Reiniciam o Host

Os aplicativos **halt**, **shutdown** e **reboot** são utilizados para desligar o sistema ou reinicializá-lo. Recomenda-se configurar as permissões de acesso a estes aplicativos, já

que sua utilização por parte de usuários não autorizados pode causar a indisponibilidade (acidental ou proposital) do sistema.

```
root@fusion:~# for BIN in halt shutdown reboot init ;do
       chmod 750 $BIN;
       chown root $BIN;
       chgrp devunix $BIN;
done

root@fusion:~#
```

5.10 – Arquivos de Configuração do Bootloader

O arquivo ***/etc/lilo.conf*** ou ***/etc/default/grub***, podem conter senhas de acesso e outras informações sensíveis. O acesso não autorizado a este arquivo pode levar ao comprometimento da integridade e da disponibilidade do sistema.

Podemos criar um grupo para dar permissão de execução às exceções da seguinte forma:

```
root@fusion:~# chmod 640 /etc/default/grub

root@fusion:~# chown root /etc/default/grub

root@fusion:~# chgrp root /etc/default/grub

root@fusion:~#
```

Observação: Deve-se observar qual gerenciador de boot (**LILO** ou **GRUB**) é utilizado no sistema, pois quando for utilizado um deles, o controle não se aplicará para o outro.

6 – Referências Bibliográficas para Proteção de Sistema de Arquivos

Referencias Bibliograficas

[1] Morimoto, Carlos. Disponível em: <http://www.hardware.com.br/termos/bios>. Acessado em: 21/07/2012.

[2] Duarte, Helto, <http://heltonduarte.com/2009/07/03/gerenciadores-de-boot/> Acessado em 21/07/2012.

[3] Ribeiro, Uira – Certificação Linux, 1º Ed, São Paulo, 2004, Axcel Books

[4] Manual do GNU GRUB v2. Disponível em: <http://www.gnu.org/software/grub/manual/html_node/Security.html#Security>. Acessado em: 24/07/2012.